PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-153029
(43)Date of publication of application : 10.06.1997

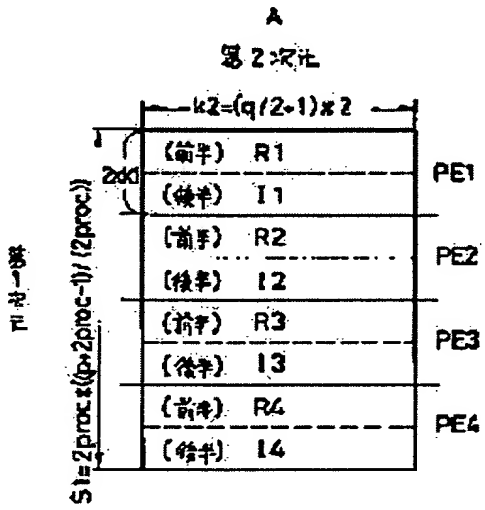
(51)Int.Cl. G06F 17/14
G06F 15/16

(21)Application number : 07-311224
(22)Date of filing : 29.11.1995

(71)Applicant : FUJITSU LTD
UNIV AUSTRALIAN NATL
(72)Inventor : NAKANISHI MAKOTO
MARKUS HOEGLUND

(54) MEMORY DISTRIBUTED PARALLEL COMPUTER FOR FAST FOURIER TRANSFORMATION AND ITS METHOD

(57)Abstract:
PROBLEM TO BE SOLVED: To efficiently execute fast Fourier transformation with a small data transfer volume.
SOLUTION: Two-dimensional real number data are stored in a two-dimensional array A with size s1 × k2 extending over four processors PE1, PE2, PE3, and PE4. The size in the first dimension (the dimension of rows) of a partial array in each processor of the array A is an even number, and each partial array is divided into two sets of row vectors of the first half and the latter half. One row vector of the first half is regarded as the real part and one row vector in the latter half is regarded as the imaginary part to execute complex Fourier transformation in each processor, and the result of real Fourier transformation for the second dimension of each row vector is taken out. Thereafter, the array A is transposed by parallel data transfer, and complex Fourier transformation for the first dimension is executed, and the result is transposed again to obtain the result of two-dimensional Fourier transformation. Fourier transformation formation for the first dimension and that for the second dimension of the array A can be executed in each processor in the closed form to efficiently perform the processing.



LEGAL STATUS

[Date of request for examination] 04.02.2000
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平9－153029

(43)公開日　平成９年(1997)６月10日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　17/14			Ｇ０６Ｆ　15/332	Ａ	
15/16	３９０		15/16	３９０Ｚ	

審査請求　未請求　請求項の数13　ＯＬ　（全 22 頁）

(21)出願番号　特願平7－311224

(22)出願日　平成７年(1995)11月29日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(71)出願人　593094659
ジ　オーストラリアン　ナショナル　ユニバシティー
オーストラリア国，エーシィーティー，0200，キャンベラ　（番地なし）

(72)発明者　中西　誠
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74)代理人　弁理士　大菅　義之　（外１名）

最終頁に続く

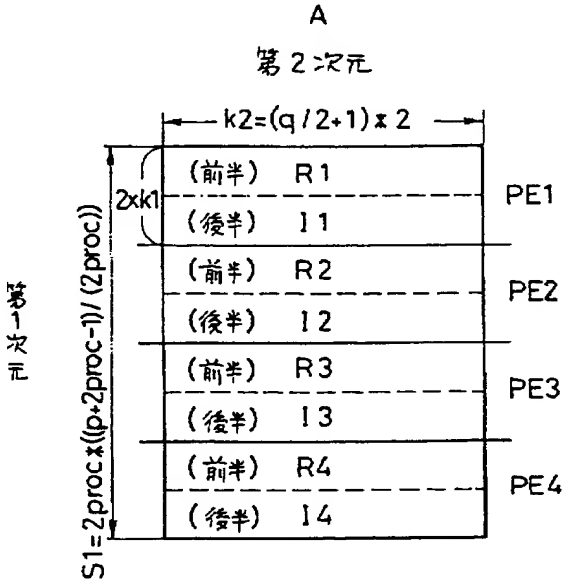
(54)【発明の名称】　高速フーリエ変換を行うメモリ分散型並列計算機およびその方法

(57)【要約】

【目的】　メモリ分散型並列計算機システムにおいて、少ないデータ転送で効率良く高速フーリエ変換を実行することを目的とする。

【構成】　２次元実数データを、４つのプロセッサＰＥ１、ＰＥ２、ＰＥ３、ＰＥ４にまたがるｓ１×ｋ２の大きさの２次元配列Ａに格納する。配列Ａの各プロセッサ内の部分配列の第１次元（行の次元）の大きさは偶数であり、各部分配列は前半と後半の２組の行ベクトルに分割される。前半の１つの行ベクトルを実部、後半の１つの行ベクトルを虚部とみなして、各プロセッサ内で複素フーリエ変換を行い、各行ベクトルの第２次元についての実フーリエ変換の結果を取り出す。その後、配列Ａを並列データ転送により転置して、第１次元についての複素フーリエ変換を行い、その結果を再び転置して２次元フーリエ変換の結果を求める。配列Ａの第１次元および第２次元の各フーリエ変換を各プロセッサ内で閉じて行うことができ、処理が効率化される。

２次元配列の分割を示す図



【特許請求の範囲】

【請求項１】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムにおいて、

実数データを、第１次元と第２次元からなる第１の２次元配列として入力する入力手段と

前記第１の２次元配列の第１次元をプロセッサ数で分割して、第１次元が偶数であるような複数の部分配列を生成し、該複数の部分配列を前記複数のプロセッサに分散して記憶する配列記憶手段と、

前記複数の部分配列の各々を前記第１次元で２つに分割して得られる第１のデータと第２のデータのうち、該第１のデータを記憶する第１の記憶手段と、

前記第２のデータを記憶する第２の記憶手段と、

前記配列記憶手段から前記第１のデータおよび第２のデータを前記第１の記憶手段および第２の記憶手段に移し、該第１のデータを実部、該第２のデータを虚部とみなして前記第２次元についての複素フーリエ変換を各プロセッサ内で行い、該複素フーリエ変換の第１の変換結果を利用して、前記実数データのフーリエ変換の第２の変換結果を求める計算手段と、

前記第２の変換結果を出力する出力手段とを備えることを特徴とする並列計算機。

【請求項２】　２次元配列を転置する転置手段をさらに備え、前記計算手段は、前記第１の２次元配列の第２次元についての実フーリエ変換の第３の変換結果の一部を、前記第１の変換結果から求めて、該第３の変換結果の一部の実部および虚部からなる第２の２次元配列を前記配列記憶手段に格納し、前記転置手段は、該配列記憶手段に格納された該第２の２次元配列を転置して前記第１の記憶手段および第２の記憶手段に格納することにより、前記第３の変換結果の一部の実部および虚部をそれぞれ該第１の記憶手段および第２の記憶手段に格納し、前記計算手段は、該第１の記憶手段および第２の記憶手段のデータを用いて、該第２の２次元配列の第１次元についての複素フーリエ変換を各プロセッサ内で行うことを特徴とする請求項１記載の並列計算機。

【請求項３】　前記転置手段は、前記第２の２次元配列をブロック単位に分割して、プロセッサ間で該ブロック単位の並列データ転送を行うことにより、該第２の２次元配列を転置することを特徴とする請求項２記載の並列計算機。

【請求項４】　前記転置手段は、前記第２の２次元配列の第１次元についての複素フーリエ変換の結果得られる第３の２次元配列を再び転置し、前記第２の変換結果として前記配列記憶手段に格納することを特徴とする請求項２記載の並列計算機。

【請求項５】　前記入力手段は、前記実数データが１次元データのとき、該１次元データを前記第１の２次元配列の形式に変換して入力することを特徴とする請求項１記載の並列計算機。

【請求項６】　２次元配列を転置する転置手段をさらに備え、前記計算手段は、前記第１の２次元配列の第２次元についての実フーリエ変換の第３の変換結果の一部を、前記第１の変換結果から求めて、該第３の変換結果の一部に回転因子を乗算し、乗算結果の実部および虚部からなる第２の２次元配列を前記配列記憶手段に格納し、前記転置手段は、該配列記憶手段に格納された該第２の２次元配列を転置して前記第１の記憶手段および第２の記憶手段に格納することにより、前記第３の変換結果の一部の実部および虚部をそれぞれ該第１の記憶手段および第２の記憶手段に格納し、前記計算手段は、該第１の記憶手段および第２の記憶手段のデータを用いて、該第２の２次元配列の第１次元についての複素フーリエ変換を各プロセッサ内で行い、前記第２の変換結果を求めることを特徴とする請求項５記載の並列計算機。

【請求項７】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムにおいて、

実数データを、第１次元と第２次元を有する第１の多次元配列として入力する入力手段と前記第１の多次元配列の第１次元をプロセッサ数で分割して、第１次元が偶数であるような複数の部分配列を生成し、該複数の部分配列を前記複数のプロセッサに分散して記憶する配列記憶手段と、

前記複数の部分配列の各々を前記第１次元で２つに分割して得られる第１のデータと第２のデータのうち、該第１のデータを記憶する第１の記憶手段と、

前記第２のデータを記憶する第２の記憶手段と、

前記配列記憶手段から前記第１のデータおよび第２のデータを前記第１の記憶手段および第２の記憶手段に移し、該第１のデータを実部、該第２のデータを虚部とみなして前記第２次元についての複素フーリエ変換を各プロセッサ内で行い、該複素フーリエ変換の第１の変換結果を利用して、前記実数データのフーリエ変換の第２の変換結果を求める計算手段と、

前記第２の変換結果を出力する出力手段とを備えることを特徴とする並列計算機。

【請求項８】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムを構成するプロセッサであって、

第１次元と第２次元を有する多次元配列の形式で入力された実数データのうち、該第１次元をプロセッサ数で分割して割り当てられた部分データを記憶するための、第１次元が偶数であるような格納領域を有する配列記憶手段と、

前記格納領域に記憶された前記部分データを前記第１次

元で２つに分割して得られる第１のデータと第２のデータのうち、該第１のデータを記憶する第１の記憶手段と、
前記第２のデータを記憶する第２の記憶手段と、
前記配列記憶手段から前記第１のデータおよび第２のデータを前記第１の記憶手段および第２の記憶手段に移し、該第１のデータを実部、該第２のデータを虚部とみなして前記第２次元についての複素フーリエ変換を行い、該複素フーリエ変換の第１の変換結果を利用して、前記実数データのフーリエ変換を求める計算手段とを備えることを特徴とするプロセッサ。

【請求項９】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムにおいて、
長さＮ＝ｐ×ｑの１次元データをｐ×ｑの第１の２次元配列とみなし、該第１の２次元配列を第１次元で分割して、各プロセッサ内の第１次元の大きさが偶数になるように分散配置する配列記憶手段と、
各プロセッサで第２次元のベクトルのペアを複素数とみなして、複素フーリエ変換を行い、変換結果から実フーリエ変換の（ｑ／２＋１）部分を計算した後、ローテーション演算を行う第１の計算手段と、
前記ローテーション演算の結果を、第１次元を分割配置した（ｑ／２＋１）×ｐの第２の２次元配列に転置する転置手段と、
該第２の２次元配列の長さｐの第２次元のベクトルに対する１次元複素フーリエ変換を各プロセッサで並列に行って、前記１次元データの実フーリエ変換の結果を求める第２の計算手段とを備えることを特徴とする並列計算機。

【請求項１０】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムにおいて、
（ｐ，ｑ）の大きさの２次元データをｐ×ｑの第１の２次元配列に入力し、該第１の２次元配列を第１次元で分割して、各プロセッサ内の第１次元の大きさが偶数になるように分散配置する配列記憶手段と、
各プロセッサで第２次元のベクトルのペアを複素数とみなして、複素フーリエ変換を行い、変換結果から実フーリエ変換の（ｑ／２＋１）部分を計算する第１の計算手段と、
前記第１の計算手段による計算結果を、第１次元を分割配置した（ｑ／２＋１）×ｐの第２の２次元配列に転置する第１の転置手段と、
該第２の２次元配列の長さｐの第２次元のベクトルに対する１次元複素フーリエ変換を各プロセッサで並列に行う第２の計算手段と、
該第２の２次元配列の変換結果をｐ×（ｑ／２＋１）の配列に転置して、前記２次元データの実フーリエ変換の結果のうち（１：ｐ，１：ｑ／２＋１）の部分を求める第２の転置手段とを備えることを特徴とする並列計算機。

【請求項１１】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムにおいて、
（ｐ，ｑ，ｒ）の大きさの３次元データをｐ×ｑ×ｒの第１の３次元配列に入力し、該第１の３次元配列を第１次元で分割して、各プロセッサ内の第１次元の大きさが偶数になるように分散配置する配列記憶手段と、
各プロセッサで第３次元のベクトルのペアを複素数とみなして、複素フーリエ変換を行い、変換結果から実フーリエ変換の（ｒ／２＋１）部分を計算した後、第２次元のベクトルに対するフーリエ変換を各プロセッサで並列に行う第１の計算手段と、
前記第１の計算手段による計算結果を、第１次元を分割配置した（ｒ／２＋１）×ｑ×ｐの第２の３次元配列に転置する第１の転置手段と、
該第２の３次元配列の長さｐの第３次元のベクトルに対する１次元複素フーリエ変換を各プロセッサで並列に行う第２の計算手段と、
該第２の３次元配列の変換結果をｐ×ｑ×（ｒ／２＋１）の配列に転置して、前記３次元データの実フーリエ変換の結果のうち（１：ｐ，１：ｑ，１：ｒ／２＋１）の部分を求める第２の転置手段と
を備えることを特徴とする並列計算機。

【請求項１２】　データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムにおける記憶媒体であって、
実数データを、第１次元と第２次元を有する第１の多次元配列として入力する入力手段と前記第１の多次元配列の第１次元をプロセッサ数で分割して、第１次元が偶数であるような複数の部分配列を生成し、該複数の部分配列を前記複数のプロセッサに分散して配置する分散配置手段と、
前記複数の部分配列の各々を前記第１次元で２つに分割して第１のデータと第２のデータを生成し、該第１のデータを記憶する第１の記憶域と該第２のデータを記憶する第２の記憶域とを割り当てる割り当て手段と、
前記第１のデータおよび第２のデータを前記第１の記憶域および第２の記憶域に移し、該第１のデータを実部、該第２のデータを虚部とみなして前記第２次元についての複素フーリエ変換を各プロセッサ内で行い、該複素フーリエ変換の第１の変換結果を利用して、前記実数データのフーリエ変換の第２の変換結果を求める計算手段と、
前記第２の変換結果を出力する出力手段とを備えること

を特徴とする記憶媒体。

【請求項１３】　メモリ分散型並列計算機システムにおいて、データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行する方法であって、

実数データを、第１次元と第２次元を有する第１の多次元配列として入力し、

前記第１の多次元配列の第１次元をプロセッサ数で分割して、第１次元が偶数であるような複数の部分配列を生成し、該複数の部分配列を前記複数のプロセッサに分散して配置し、

前記複数の部分配列の各々を前記第１次元で２つに分割して第１のデータと第２のデータを生成し、

該第１のデータおよび第２のデータを第１の記憶域および第２の記憶域に移し、

該第１のデータを実部、該第２のデータを虚部とみなして前記第２次元についての複素フーリエ変換を各プロセッサ内で行い、

該複素フーリエ変換の第１の変換結果を利用して、前記実数データのフーリエ変換の第２の変換結果を求めることを特徴とするフーリエ変換処理方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は高速フーリエ変換を行うメモリ分散型並列計算機システム、およびフーリエ変換処理方法に関する。

【０００２】

【従来の技術とその問題点】今日、科学技術計算等において、大規模な高速フーリエ変換（ＦＦＴ：Fast Fourier Transformation）を計算するために、並列計算機システムがよく用いられている。ＦＦＴは、離散型フーリエ変換における項数がいくつかの因数に分解できる時、その性質を利用して必要な演算回数を削減した計算アルゴリズムである。特に、大規模な実フーリエ変換をＶＰＰ５００等のベクトル分散並列計算機で行うとき、その性能を最大限に引き出すために、データ転送が少なく一括して計算できる方法が望まれている。

【０００３】従来の並列計算機によるＦＦＴ処理においては、フーリエ変換の度に、複数のプロセッシング・エレメント（以後、ＰＥ、またはプロセッサと記す）に分散したデータをアクセスする必要がある。このため、多量のデータ転送が行われ、処理効率が低下するという問題が生じる。

【０００４】実数のフーリエ変換を行うには、複素フーリエ変換の虚部を０として計算する方法と、２つの実数をそれぞれ複素数の実部、虚部とみなして、複素フーリエ変換を行う方法の２つがある。後者は、前者に比べて使用するメモリ量が約半分で済むという利点がある。これは、フーリエ変換した結果の複素数の中に共役関係を持つペアがあり、求めるべきデータを約半分に減らすことができることを利用したためである。

【０００５】実際には、変換される実数を複数の列に並べて、偶数列と奇数列（または偶数行と奇数行）のデータをそれぞれ実部および虚部とみなして複素フーリエ変換を行っており、最後に、その結果から必要な約半分の実フーリエ変換の結果を作り出している。残りの半分の結果は、共役関係を利用して求めることができる。しかしながら、偶数列と奇数列（または偶数行と奇数行）のデータは必ずしも同じプロセッサに格納されるとは限らず、フーリエ変換のアクセスパターンが全プロセッサにまたがるため、多量のデータ転送が必要となる。

【０００６】並列計算機においてはデータ転送に伴うオーバヘッドが大きいため、このような多量のデータ転送が生じると、ベクトル処理および並列処理の効率が悪くなる。

【０００７】本発明は、少ないデータ転送で効率良く実ＦＦＴを実行することのできるメモリ分散型並列計算機システムと、その方法を提供することを目的とする。

【０００８】

【問題を解決するための手段】図１は、本発明の並列計算機システムの原理図である。図１の並列計算機システムは、データを複数のプロセッサのメモリに分散配置し、プロセッサ間でデータ転送を行いながら高速フーリエ変換を実行することのできるメモリ分散型並列計算機システムであって、入力手段１、配列記憶手段２、第１の記憶手段３、第２の記憶手段４、計算手段５、転置手段６、および出力手段７を備える。

【０００９】入力手段１は、実数データを、第１次元と第２次元からなる第１の２次元配列として入力する。配列記憶手段２は、第１の２次元配列の第１次元をプロセッサ数で分割して、第１次元が偶数であるような複数の部分配列を生成し、それらの複数の部分配列を上記複数のプロセッサに分散して記憶する。

【００１０】第１の記憶手段３は、上記複数の部分配列の各々を第１次元で２つに分割して得られる第１のデータと第２のデータのうち、第１のデータを記憶し、第２の記憶手段４は第２のデータを記憶する。

【００１１】計算手段５は、配列記憶手段２から第１のデータおよび第２のデータを第１の記憶手段３および第２の記憶手段４に移し、第１のデータを実部、第２のデータを虚部とみなして第２次元についての複素フーリエ変換を各プロセッサ内で行い、その複素フーリエ変換の第１の変換結果を求める。そして、第１の変換結果を利用して、上記実数データのフーリエ変換の第２の変換結果を求める。

【００１２】出力手段７は、上記第２の変換結果を出力する。また、計算手段５は、上記第１の２次元配列の第２次元についての実フーリエ変換の第３の変換結果の一部を、上記第１の変換結果から求めて、その第３の変換結果の一部の実部および虚部からなる第２の２次元配列

を配列記憶手段２に格納する。このとき、転置手段６は、配列記憶手段２に格納された第２の２次元配列を転置して第１の記憶手段３および第２の記憶手段４に格納することにより、上記第３の変換結果の一部の実部および虚部をそれぞれ第１の記憶手段３および第２の記憶手段４に格納する。そして、計算手段５は、第１の記憶手段３および第２の記憶手段４のデータを用いて、第２の２次元配列の第１次元についての複素フーリエ変換を各プロセッサ内で行い、その結果から上記第２の変換結果を求める。

【００１３】例えば、図１の入力手段１と出力手段７は、実施例における図２の入出力装置１３に対応し、配列記憶手段２、第１の記憶手段３、第２の記憶手段４は、各プロセッサ１１－１、１１－２、・・・、１１－M内のメモリ１６に対応する。また、計算手段５と転置手段６は、各プロセッサ１１－１、１１－２、・・・、１１－M内の処理部１５に対応する。さらに、例えば、これらの各手段の機能をプログラム化して記憶したディスク装置等の記憶媒体もまた、これらの各手段に対応する。

【００１４】

【作用】入力手段１が入力した２次元実数データが、第１の２次元配列として、配列記憶手段２により複数のプロセッサに分散配置される。２次元実フーリエ変換を行うには、この第１の２次元配列を第１次元および第２次元についてそれぞれフーリエ変換すればよい。このとき、第１の２次元配列の第１次元の大きさを、例えば（２×プロセッサ数）の倍数とし、それをプロセッサ数で分割したときに、各プロセッサ内の部分配列の第１次元が偶数になるようにしておく。これにより、各部分配列を第１次元で２つに分割して格納することが可能になる。第１の記憶手段３は、各プロセッサ毎に、各部分配列を分割した２つデータのうち第１のデータを記憶し、第２の記憶手段４は第２のデータを記憶する。

【００１５】計算手段５は、第１の記憶手段３内の第１のデータを複素数の実部とみなし、第２の記憶手段４内の第２のデータを虚部とみなして、各プロセッサ内で複素数のフーリエ変換を行うことができる。これにより、分割されていない第２次元についての複素フーリエ変換が各プロセッサ内で行われ、第１の変換結果が得られる。第１のデータおよび第２のデータのそれぞれの第２次元についての実フーリエ変換の結果は、第１の変換結果から容易に得ることができ、次に第１次元についての複素フーリエ変換を行えば、上記実数データの２次元フーリエ変換の結果である第２の変換結果が得られる。

【００１６】このように、入力された２次元データを各プロセッサ内で２つの部分に分け、一方を実部、他方を虚部とみなすことにより、２組の実数データをまとめてフーリエ変換することができる。このとき、計算は各プロセッサ内で閉じて行われるため、データ転送に要する時間が節約され、処理が効率化される。

【００１７】尚、入力データが１次元実数データの場合は、これを２次元データに変換して配列記憶手段２に記憶させることにより、２次元データと同様に処理することが可能である。この場合は、第１次元についての複素フーリエ変換を行う前に、各プロセッサ内でローテーションの計算を行えばよい。また、３次元以上の次元の実数データのフーリエ変換についても、基本的には２次元データと同様の手法により処理することができる。

【００１８】また、第１の変換結果から第２の変換結果を得る時、計算手段５は、まず第１の２次元配列の第２次元についての実フーリエ変換の結果である第３の変換結果を、第１の変換結果から求める。このとき、第３の変換結果には一般に特定の共役関係があり、これを利用すれば、第３の変換結果の約半分は他の部分から計算できることが知られている。したがって、第３の変換結果については、その一部分（約半分）のみを求めればよい。求めた第３の変換結果の一部の実部および虚部は、第２の２次元配列として配列記憶手段２に格納される。

【００１９】しかし、このままでは、第２の２次元配列の第１次元が複数のプロセッサにまたがって配置されているため、残された第１次元についてのフーリエ変換を各プロセッサ内で閉じて行うことができない。そこで、転置手段６が第２の２次元配列を転置して、第１の記憶手段３および第２の記憶手段４に格納する。これにより、第３の変換結果の一部の実部および虚部が、それぞれ第１の記憶手段３および第２の記憶手段４に格納される。計算手段は、これらを１組の複素数として、第２の２次元配列の第１次元についての複素フーリエ変換を各プロセッサ内で行い、最終的に第２の変換結果を求める。得られた第２の変換結果は、入力された２次元実数データのフーリエ変換の結果を与えている。

【００２０】このように、第２次元についてのフーリエ変換の後に２次元配列を転置すれば、第１次元についてのフーリエ変換も複数のプロセッサで並列に処理することができる。１次元実数データの場合は、配列記憶手段２に格納された第２の２次元配列に対してローテーションの計算が行われ、その結果が転置されて、第１次元についてのフーリエ変換が行われる。

【００２１】

【実施例】以下、図面を参照しながら本発明の実施例を詳細に説明する。図２は、実施例における並列計算機システムの構成図である。図２の並列計算機システムは、ネットワーク１２により結合された複数のプロセッサ１１－１、１１－２、・・・、１１－Mからなり、ネットワーク１２に接続された入出力装置１３を備える。ネットワーク１２は、任意の２つのプロセッサ間でデータ転送を行えるように設計されており、例えばクロスバー・ネットワークである。入出力装置１３は、例えばディスプレイやキーボードを備えた端末装置であり、フーリエ

変換されるデータを入力し、変換結果を出力する。

【００２２】また、各プロセッサ１１－１、１１－２、・・・、１１－Mは、それぞれ通信部１４、処理部１５、メモリ１６、およびそれらを接続する内部バス１７を備える。通信部１４は、ネットワーク１２を介して、他のプロセッサや入出力装置１３との間でデータ転送を行う。メモリ１６は、各プロセッサに割り当てられたデータや他のプロセッサから転送されたデータを記憶する。処理部１５は、メモリ１６に記憶されたデータを用いて、演算等の処理を行う。

【００２３】本実施例においては、図２の並列計算機システムにより行われる１次元、２次元、および３次元のフーリエ変換処理について順に説明する。最初に、フーリエ変換に必要となる数学的な前提について述べる。

【００２４】１次元の離散フーリエ変換は次式で与えられる。

【００２５】

【数１】

$$f_j = \sum_{n=0}^{N-1} x_n \, \omega_N^{nj} \qquad \cdots (1)$$

【００２６】ただし、

$$\omega_N \equiv \exp(-2\pi i / N) \qquad \cdots (2)$$

とする。今、N＝p・qと因数分解できるとすると、

$$n = n1 + n2 \cdot p, \qquad \cdots (3)$$

（n1＝0，1，．．．，p－1，n2＝0，1，．．．，q－1）

$$j = j1 + j2 \cdot q, \qquad \cdots (4)$$

（j1＝0，1，．．．，q－1，j2＝0，1，．．．，p－1）

$$x(n1, n2) = x_n, \qquad \cdots (5)$$

$$f(j1, j2) = f_j \qquad \cdots (6)$$

とおくことができる。（３）、（４）、（５）、（６）式を（１）式に代入して、nについての和をn１とn２についての和に書き換えると、

【００２７】

【数２】

$$f(j1, j2) = \sum_{n1=0}^{p-1} \omega_p^{n1 \cdot j2} \omega_N^{n1 \cdot j1} \cdot \sum_{n2=0}^{q-1} x(n1, n2) \, \omega_q^{n2 \cdot j1} \qquad \cdots (7)$$

【００２８】となる。このように、離散フーリエ変換の項数Nが適当な因数に分解できるときには、１次元のデータを２次元データとみなして処理することができる。並列計算機による処理においては、例えばx（n１，n２）の第１次元を複数のプロセッサに分割して配置する。このとき、（７）式は次の４段の処理に分けられる。

【００２９】

【数３】

$$(1段)\quad Y1(n1,\ j1)=\sum_{n2=0}^{Q-1}x(n1,\ n2)\ \omega_Q^{n2j1}$$

…（8）

$$(2段)\quad Y2(n1,\ j1)=\omega_N^{n1j1}Y1(n1,\ j1)$$

…（9）

$$(3段)\quad Y3(j1,\ n1)=Y2(n1,\ j1)$$

…（10）

$$(4段)\quad Y4(j1,\ j2)=\sum_{n1=0}^{P-1}Y3(j1,\ n1)\ \omega_P^{n1j2}$$

…（11）

【0030】（8）式は2次元配列x（n1，n2）の添字n2についてのフーリエ変換を表し、（9）式はY1（n1，j1）に回転因子ω_N^{n1j1}を乗算する計算（ローテーション）を表し、（10）式は2次元配列Y2（n1，j1）の転置を表し、（11）式はY3（j1，n1）の添字n1についてのフーリエ変換を表す。第3段で転置を行うのは、常に2次元配列の第2次元についてのフーリエ変換を行うようにするためである。ここでは、2次元配列の第2次元は分割されていないので、第2次元についてのフーリエ変換は各プロセッサ内で独立に行うことができる。

【0031】次に、実1次元フーリエ変換を複素フーリエ変換と組み合わせて行う方法について説明する。変換すべき実データを$\{x1_n\}$と$\{x2_n\}$（n＝0，．．．，Q－1）とに分け、それぞれを複素数の実部と虚部とみなして、

$$z_n=x1_n+i\,x2_n$$

…（12）

とおく。また、z_n、$x1_n$、$x2_n$のフーリエ変換の結果を、それぞれ$\{\alpha z_j\}$、$\{\alpha x1_j\}$、$\{\alpha x2_j\}$とおくと、

【0032】

【数4】

$$\alpha z_j=\sum_{n=0}^{Q-1}z_n\ \omega_Q^{nj}$$

…（13）

$$\alpha x1_j=\sum_{n=0}^{Q-1}x1_n\ \omega_Q^{nj}$$

…（14）

$$\alpha x2_j=\sum_{n=0}^{Q-1}x2_n\ \omega_Q^{nj}$$

…（15）

【0033】となる。（14）、（15）式の$\alpha x1_j$、$\alpha x2_j$は、（13）式のαz_jを用いて、次のように表される。

【0034】

【数5】

$$\alpha x1_j=(\alpha z_j+(\alpha z_{Q-j})^*)/2$$

…（16）

$$\alpha x2_j=(\alpha z_j-(\alpha z_{Q-j})^*)/2i$$

…（17）

$$\alpha x1_0=Re(\alpha z_0)$$

…（18）

$$\alpha x2_0=Im(\alpha z_0)$$

…（19）

【0035】ここで、α^*はαの複素共役を表し、Re（α）、Im（α）はそれぞれαの実部、虚部を表す。

したがって、$\{\alpha^{z}{}_{j}\}$ が求まれば、（１６）、（１７）式に従って $\{\alpha^{x1}{}_{j}\}$、$\{\alpha^{x2}{}_{j}\}$ を計算すること

$$\alpha^{x1}{}_{Q-j} = (\alpha^{x1}{}_{j})^{*}, \quad j=1,\ldots,Q-1 \qquad \cdots(20)$$

$$\alpha^{x2}{}_{Q-j} = (\alpha^{x2}{}_{j})^{*}, \quad j=1,\ldots,Q-1 \qquad \cdots(21)$$

なる共役関係があるため、すべてのｊについて $\alpha^{x1}{}_{j}$、$\alpha^{x2}{}_{j}$ を求める必要はない。実際には、最初の（Ｑ／２＋１）個の $\alpha^{x1}{}_{j}$ と $\alpha^{x2}{}_{j}$ について、実フーリエ変換の結果を求めれば充分である。ただし、Ｑ／２はＱを２で割った時の整数商（剰余は切り捨て）を表す。以下、特に断らない限り、数式に現れる除算は整数商を表すものとする。

【００３６】以上の知識を前提として、１次元実ＦＦＴの方法を説明する。図３は、Ｎ＝ｐ・ｑと分解できるＮ個の実数を（１）式に従って変換する１次元実ＦＦＴのフローチャートである。図３において処理が開始される

$$s1=2\times proc\times k1,$$

$$k1=(p+2\times proc-1)/(2\times proc) \qquad \cdots(22)(23)$$

と定め、第１次元をｐｒｏｃ台のプロセッサにより均等に分割したとき、各プロセッサに格納される部分の行数

$$k2=(q/2+1)\times 2 \qquad \cdots(24)$$

以上とする。図４では、Ａの第２次元の大きさはｋ２となっており、配列ＡはＡ（ｓ１，ｋ２）と表される。次に、

$$k3=((q/2+proc)/proc)\times proc \qquad \cdots(25)$$

として、Ａの転置用の格納配列Ｂ（２×ｋ３，ｓ１）を用意する。Ｂは同じ大きさの２つの配列ＢＲ（ｋ３，ｓ１）とＢＩ（ｋ３，ｓ１）からなる。ＢＲ、ＢＩの第１次元の大きさｋ３は、Ａの第２次元の大きさｋ２の半分

$$2\times k3 \geqq k2=(q/2+1)\times 2 \qquad \cdots(26)$$

なる関係がある。

【００３８】そして、Ａ（ｓ１，ｋ２）、ＢＲ（ｋ３，ｓ１）、ＢＩ（ｋ３，ｓ１）の第１次元を各プロセッサで均等に分割し、分割された各プロセッサの部分をａ（２×ｋ１，ｋ２）、ｂｒ（ｋ３／ｐｒｏｃ，ｓ１）、ｂｉ（ｋ３／ｐｒｏｃ，ｓ１）とする。

【００３９】次に、各プロセッサ内で、（８）式に対応する１段目の処理を並列に行う。ただし、このときメモリの使用領域を節約するために、各プロセッサに割り当てられた行ベクトルを前半と後半の２組に分け、一方を実部、他方を虚部として、複素数のフーリエ変換を行う。

【００４０】まず、各プロセッサは行ベクトルのペアを作り、一方を実部、他方を虚部として別々の領域に格納する（ステップＳ２）。具体的には、格納領域ｂｒ、ｂｉをそれぞれ格納配列ｂｒ（ｋ１，２×ｋ３）、ｂｉ（ｋ１，２×ｋ３）として利用し、ａ（１：ｋ１，ｋ２）をｂｒ（ｋ１，１：ｋ２）にコピーし、ａ（ｋ１＋１：２×ｋ１，ｋ２）をｂｉ（ｋ１，１：ｋ２）にコピーする。

【００４１】図４では、４台のプロセッサＰＥ１、ＰＥ２、ＰＥ３、ＰＥ４の各々に割り当てられたａ（２×ｋ１，ｋ２）が、前半部分と後半部分に分割されている。ＰＥ１のデータはＲ１とＩ１に分けられ、ＰＥ２のデータはＲ２とＩ２に分けられ、ＰＥ３のデータはＲ３とＩ３に分けられ、ＰＥ４のデータはＲ４とＩ４に分けられる。そして、これらのデータが、図５に示すように、配列ＢＲ、ＢＩにコピーされる。図５において、Ｒ１、Ｒ２、Ｒ３、Ｒ４は実部としてＢＲに格納され、Ｉ１、Ｉ２、Ｉ３、Ｉ４は虚部としてＢＩに格納されている。このとき、ＢＲ、ＢＩは、それぞれＢＲ（ｓ１／２，２×ｋ３）、ＢＩ（ｓ１／２，２×ｋ３）なる配列として利用される。

【００４２】次に、ｂｒ（ｋ１，１：ｋ２）の行ベクトルと、対応するｂｉ（ｋ１，１：ｋ２）の行ベクトルとを束ねて、各プロセッサで複素数のフーリエ変換を行う（ステップＳ３）。つまり、ｂｒ（ｋ１，１：ｋ２）の１つの行ベクトルの各要素を（１２）式の $x1_{n}$ とみなし、対応するｂｉ（ｋ１，１：ｋ２）の行ベクトルの要素を（１２）式の $x2_{n}$ とみなして、複素数 z_{n} に対するフーリエ変換を（１３）式により計算する。ただし、このときＱ＝ｑとする。

【００４３】そして、（１６）～（１９）式により、最初の（ｑ／２＋１）個の実フーリエ変換の結果を求め、

その実部および虚部をそれぞれａ（２×ｋ１，１：ｑ／２＋１）およびａ（２×ｋ１，ｑ／２＋２：ｋ２）に格納する（ステップＳ４）。図６は、このときの格納方法を示している。図６において、Ａの第２次元の前半が実部に割り当てられ、後半が虚部に割り当てられていることが分かる。例えばプロセッサＰＥ１内では、図４のＲ１部分の行ベクトルの実フーリエ変換結果のうち、最初の（ｑ／２＋１）個が実際に求められ、それらの実部がＣＲ１に格納され、虚部がＤＲ１に格納される。また、図４のＩ１部分の行ベクトルの実フーリエ変換結果のうち、最初の（ｑ／２＋１）個が実際に求められ、それらの実部がＣＩ１に格納され、虚部がＤＩ１に格納される。他のプロセッサの領域ＣＲ２、ＤＲ２、ＣＩ２、ＤＩ２、ＣＲ３、ＤＲ３、ＣＩ３、ＤＩ３、ＣＲ４、ＤＲ４、ＣＩ４、ＤＩ４についても同様である。

【００４４】図４、図５のような格納方法によれば、変換される複素数の実部と虚部が同じプロセッサ内にあるので、複素フーリエ変換を各プロセッサ内で独立に行うことができ、余分なデータ転送を行う必要がない。また、長さｑの実データのペアに対して、一方を実部、他方を虚部とみなして複素フーリエ変換を行った結果から、（ｑ／２＋１）個の実フーリエ変換の結果を計算する方法をベースにして、１次元のフーリエ変換を束ねて行うことで、プロセッサのベクトル性能を引き出すことができる。

【００４５】次に、各プロセッサ内で（９）式に対応する２段目の処理を並列に行って、ローテーションを計算する（ステップＳ５）。この計算結果は再び配列Ａに格納される。

【００４６】この後、（１０）式に対応する３段目の処理を行って、配列Ａのｐ×（ｑ／２＋１）×２のデータをブロック（小領域）に分割し、ブロックレベルで転置して配列Ｂに格納する（ステップＳ６）。このとき、格納領域ＢＲ、ＢＩはそれぞれ配列ＢＲ（ｋ３，ｓ１）、ＢＩ（ｋ３，ｓ１）として利用され、その第１次元に関して分割されている。そして、プロセッサ間のデータ転送により、Ａ（ｐ，１：ｑ／２＋１）のデータがＢＲ（１：ｑ／２＋１，ｐ）に転置され、Ａ（ｐ，ｑ／２＋２：（ｑ／２＋１）×２）のデータがＢＩ（１：ｑ／２＋１，ｐ）に転置される。

【００４７】図７は、ステップＳ６で行われる行列の転置処理のフローチャートである。図７を参照しながら、配列Ａの実部を配列ＢＲに転置する処理について説明する。図７において処理が開始されると、まず配列Ａの実部、配列ＢＲをメッシュで区切る（ステップＳ１１）。これにより、各プロセッサ内にあるＡの実部とＢＲの部分が、それぞれプロセッサ数ｐｒｏｃに相当する数のブロックに分割される。図８は、転置前の配列Ａの実部（または虚部）の分割例を示しており、図９は、転置後のデータを格納する配列ＢＲ（またはＢＩ）の分割例を示している。

【００４８】図８において、Ａの実部のうちプロセッサＰＥ１内にあるデータは、A_{11}、A_{12}、A_{13}、A_{14}の４つのブロックに分割されている。同様に、プロセッサＰＥ２内のデータはブロックA_{21}、A_{22}、A_{23}、A_{24}に分割され、プロセッサＰＥ３内のデータはブロックA_{31}、A_{32}、A_{33}、A_{34}に分割され、プロセッサＰＥ４内のデータはブロックA_{41}、A_{42}、A_{43}、A_{44}に分割される。また、図９において、ＢＲのうちプロセッサＰＥ１内にあるデータは、B_{11}、B_{12}、B_{13}、B_{14}の４つのブロックに分割されている。同様に、プロセッサＰＥ２内のデータはブロックB_{21}、B_{22}、B_{23}、B_{24}に分割され、プロセッサＰＥ３内のデータはブロックB_{31}、B_{32}、B_{33}、B_{34}に分割され、プロセッサＰＥ４内のデータはブロックB_{41}、B_{42}、B_{43}、B_{44}に分割される。

【００４９】次に、各プロセッサで、Ｋ＝（そのプロセッサの番号）、Ｊ＝Ｋ、＃ｃｔ＝１とおき（ステップＳ１２）、ブロックA_{JK}の転置行列A_{JK}^{T}を求めて、ブロックB_{KJ}に格納する（ステップＳ１３）。ここで、B_{KJ}が同じプロセッサ内にあればA_{JK}^{T}をそこに格納し、B_{KJ}が他のプロセッサ内にあるときはデータ転送を行う。そして、Ｋ＝ｍｏｄ（Ｋ，ｐｒｏｃ）＋１、＃ｃｔ＝＃ｃｔ＋１とおき（ステップＳ１４）、＃ｃｔとｐｒｏｃを比較する（ステップＳ１５）。ここで、ｍｏｄ（Ｋ，ｐｒｏｃ）は、Ｋをｐｒｏｃで割った時の剰余を意味する。＃ｃｔがｐｒｏｃを超えていなければステップＳ１３以降の処理を繰り返し、＃ｃｔがｐｒｏｃを超えると処理を終了する。

【００５０】例えば、プロセッサＰＥ１の場合は、まず最初にA_{11}^{T}を求めて同じプロセッサ内のB_{11}に格納する（ステップＳ１３）。次に、Ｋ＝ｍｏｄ（１，４）＋１＝２となるので（ステップＳ１４）、A_{12}^{T}を求める（ステップＳ１３）。ところが、対応する格納先のB_{21}はプロセッサＰＥ２内にあるため、A_{12}^{T}をプロセッサＰＥ２に転送する。同様にして、A_{13}^{T}、A_{14}^{T}を順次求めて、プロセッサＰＥ３のB_{31}、ＰＥ４のB_{41}にそれぞれ転送する。すべてのプロセッサが同様の処理を行った結果、ＢＲの内容は図１０に示すようになる。図１０において、プロセッサＰＥ１内にはA_{11}^{T}、A_{21}^{T}、A_{31}^{T}、A_{41}^{T}が格納され、プロセッサＰＥ２内にはA_{12}^{T}、A_{22}^{T}、A_{32}^{T}、A_{42}^{T}が格納され、プロセッサＰＥ３内にはA_{13}^{T}、A_{23}^{T}、A_{33}^{T}、A_{43}^{T}が格納され、プロセッサＰＥ４内にはA_{14}^{T}、A_{24}^{T}、A_{34}^{T}、A_{44}^{T}が格納されている。

【００５１】ステップＳ１３においては、ｐｒｏｃ台のプロセッサの間でデータ転送が発生するが、ＶＰＰ５００のように各プロセッサに対して同時に読込みと書込みができる並列計算機では、配列の対角方向に並ぶブロック要素について並列にデータ転送を行うことができる。例えば、図８の斜線部分のブロックA_{12}、A_{23}、A_{34}、

A_{41}の転置データA_{12}^T、A_{23}^T、A_{34}^T、A_{41}^Tは、図9の斜線部分の各ブロックに並列に転送される。このような並列転置処理を行うことにより、データ転送のコストが軽減される。

【0052】配列Aの虚部を配列BIに転置する場合も、同様の処理を行う。このとき、Aの虚部は図8のように分割され、BIは図9のように分割される。ここで、配列Aを転置しておくことにより、配列Aの列ベクトルに対するフーリエ変換を配列Bの行ベクトルに対するフーリエ変換に置き換えることができ、次の4段目の処理を各プロセッサで並列に行うことができる。

【0053】そして、各プロセッサで（11）式に対応する4段目の処理を並列に行って（ステップS7）、処理を終了する。ステップS7では、各プロセッサは行列の転置結果を用いて、メモリ16内の長さpの各行ベクトルに対する複素FFTを行う。これにより、与えられたN＝p×q個の実数のフーリエ変換の結果のうち、p×（q／2＋1）個が得られる。残りのp×（q－（q／2＋1））個の実フーリエ変換の値は得られた結果と共役関係にあるため、これですべての結果を求めたことになる。

【0054】次に、図11から図16までを参照しながら、1次元実FFTの具体例について説明する。図11は、入力データを2次元の格納配列に収容して4台のプロセッサに分散配置した例を示している。図11では、N＝300個の実数が1次元データとして入力され、p＝20、q＝15として、20×15個の2次元データx（n1，n2）（n1＝1，2，．．．，20，　n2＝1，2，．．．，15）として分散配置される（ステップS1）。このとき、（22）、（23）、（24）、（25）式よりs1＝24、k1＝3、k2＝16、k3＝8となり、20×15個のデータを格納する配列Aの大きさはs1×k2＝24×16となる。また、Aの第1次元を4台のプロセッサで分割するので、1つのプロセッサに割り当てられる格納配列の大きさは6×16となり、その行ベクトルの数（6）は偶数になる。ここでは、プロセッサPE1、PE2、PE3のすべての行ベクトルとプロセッサPE4の2本の行ベクトルにデータが格納されている。そして、配列Aのデータを格納しない部分の要素（図11の＊印）の値は、必要のないオーバフローやアンダーフローを避けるために0にしておく。

【0055】こうして配列Aに格納されたデータは、各プロセッサ内で別の格納配列Bにコピーされる（ステップS2）。図12は、図11のデータがコピーされた配列Bを示している。ここでは、配列Bはs1×（2×k3）＝24×16の配列として利用され、それぞれ12×16の大きさを持つ配列BRとBIに分けられる。そして、それらのうちのそれぞれ3×16の部分が各プロセッサに配置される。各プロセッサに割り当てられたAの6本の行ベクトルのうち、上半分の3本はBRの対応する領域にコピーされ、下半分の3本はBIの対応する領域にコピーされる。

【0056】次に、各プロセッサ内で、BRに格納されたデータを実部とみなし、BI内でそれと同じ位置に格納されたデータを虚部とみなして複素数をつくり、そのフーリエ変換を計算する（ステップS3）。例えば、プロセッサPE1内の1行目の行ベクトルについては、$x1_0$＝x（1，1），$x1_1$＝x（1，2），．．．，$x1_{14}$＝x（1，15）、かつ、$x2_0$＝x（4，1），$x2_1$＝x（4，2），．．．，$x2_{14}$＝x（4，15）として、（12）式に代入し、15個の複素数z_nを作る。そして、Q＝15として（13）式によりそのフーリエ変換を求め、その結果得られる複素数$\alpha^{z}{}_{j}$（j＝0，1，2，．．．，14）の実部をBRに格納し、虚部をBIに格納する。他の行ベクトルについても同様である。

【0057】図13は、フーリエ変換の結果を格納した配列BR、BIを示している。図13において、BR内のX（n1，n2）（n1＝1，2，3，7，8，9，13，14，15，19，20，　n2＝1，．．．，15）は、各フーリエ変換結果の実部を表し、BI内のX（n1，n2）（n1＝4，5，6，10，11，12，16，17，18，　n2＝1，．．．，15）は、各フーリエ変換結果の虚部を表している。例えば、図12のx（1，n2）を実部、x（4，n2）を虚部として変換した結果得られた複素数の実部がX（1，n2）、虚部がX（4，n2）である。

【0058】次に、BR、BIに格納された複素フーリエ変換の結果を利用して、（16）～（19）式により実フーリエ変換の結果$\alpha^{x1}{}_{j}$、$\alpha^{x2}{}_{j}$を求め、配列Aに格納する（ステップS4）。ただし、この場合は（20）、（21）式の共役関係があるので、jの最初の15／2＋1＝8個の値について$\alpha^{x1}{}_{j}$、$\alpha^{x2}{}_{j}$を求めればよい。こうして求められたx（n1，n2）のn2に関するフーリエ変換の結果をα（n1，j2）（n1＝1，．．．，20，　j2＝1，．．．，8）と書くことにする。例えば、x（1，1），x（1，2），．．．，x（1，15）のフーリエ変換の結果のうち、約半分に当たるα（1，1），α（1，2），．．．，α（1，8）が求められ、同様に、x（4，1），x（4，2），．．．，x（4，15）のフーリエ変換の結果のうち、約半分に当たるα（4，1），α（4，2），．．．，α（4，8）が求められる。これらの実フーリエ変換の結果は、図14に示すように配列Aに格納される。図14において、α（n1，j2）の実部Re（α（n1，j2））はAの左半分に格納され、虚部Im（α（n1，j2））はAの右半分に格納されている。

【0059】そして、各プロセッサ内で、（9）式に準

じてα（n1，j2）に回転因子を乗算し、その結果をα′（n1，j2）として再び配列Aに格納する（ステップS5）。図15は、α′（n1，j2）を格納した配列Aを示している。図15においては、図14の場合と同様に、α′（n1，j2）の実部Re（α′（n1，j2））はAの左半分に格納され、虚部Im（α′（n1，j2））はAの右半分に格納されている。

【0060】次に、プロセッサPE1～PE4は、行列Aを図7の処理によりブロックレベルで転置して、配列BR、BIに格納する（ステップS6）。このとき、Aの左半分と右半分はそれぞれ図8に示すように分割され、対角方向に並ぶ4つのブロックが転置されて並列転送される。このような並列転送を繰り返すことにより、行列全体が転置される。図16は、転置後のデータを格納した配列BR、BIを示している。ここでは、BR、BIはそれぞれ8×24の配列として利用され、各プロセッサはBR、BIのそれぞれの2×24の部分を格納している。例えば、図15のプロセッサPE1の1行目の行ベクトルRe（α′（1，1）），．．．，Re（α′（1，8）），Im（α′（1，1）），．．．，Im（α′（1，8））のうち、Re（α′（1，1）），．．．，Re（α′（1，8））はBRの1列目に格納され、Im（α′（1，1）），．．．，Im（α′（1，8））はBIの1列目に格納される。

【0061】次に、各プロセッサ内で転置後の行ベクトルを使用して、α′（n1，j2）（n1＝1，．．．，20）のn1に関するフーリエ変換を計算する（ステップS7）。このときの計算方法は、ステップS3の場合と同様である。計算に必要なα′（n1，j2）の実部と虚部は、それぞれBRとBI内の対応する位置に格納されている。そして、各行ベクトルについて得られた20個のフーリエ変換の結果は、実部と虚部に分けてそれぞれBRとBIに格納される。こうして、最終的に、図11の20×15個の実数に対する1次元実FFTの結果のうち、最初の約半分が求められる。残りの部分は、得られた結果の複素共役を取ることにより求めることができる。

【0062】次に、2次元実FFTの方法について説明する。p×qの2次元データの離散フーリエ変換は次式で与えられる。

【0063】

【数6】

$$f_{j1j2}=\sum_{n2=0}^{q-1}\sum_{n1=0}^{p-1}\omega_q^{n2j2}\,\omega_p^{n1j1}\,x_{n1n2}\qquad\cdots(27)$$

【0064】（27）式を（7）式と比べると、回転因子の乗算を除いて右辺の計算手順が同じであることが分かる。ただし、x_{n1n2}は（7）式のx（n1，n2）に対応し、f_{j1j2}はf（j2，j1）に対応している。したがって、並列計算機により（27）式を計算するには、図3から3段目の処理を除いた処理を行い、得られた2次元配列を転置すればよい。

【0065】図17は、p×q個の2次元データを（27）式に従って変換する2次元実FFTのフローチャートである。図17において処理が開始されると、まず入出力装置13は与えられたp×qの2次元データを取り込み、これをp×qの2次元配列として、各プロセッサに所定数の行ベクトルを均等に割り当てる（ステップS21）。割当てられた行ベクトルは、図4と同様にして、各プロセッサのメモリ16に格納される。

【0066】次に、各プロセッサ内で、1次元実FFTの場合と同様にして、（8）式に対応する1段目の処理を並列に行う。まず、各プロセッサは行ベクトルのペアを作り、一方を実部、他方を虚部として、図5と同様の格納領域BR、BIにそれぞれコピーする（ステップS22）。次に、各プロセッサで、BRの行ベクトルと、対応するBIの行ベクトルとを束ねて、（13）式により複素数のフーリエ変換を行う（ステップS23）。そして、その結果から（16）～（19）式により、最初の（q／2＋1）個の実フーリエ変換の結果を求め、その実部および虚部を、図6と同様にして配列Aに格納する（ステップS24）。

【0067】次に、2段目の処理を行って、配列Aのp×（q／2＋1）×2のデータをブロック（小領域）に分割し、ブロックレベルで転置して配列Bに格納する（ステップS25）。このときの転置処理の方法は、図7～10に示す方法と同様である。

【0068】そして、各プロセッサで3段目の処理を並列に行って、行列の転置結果における長さpの各行ベクトルに対する複素FFTを計算する（ステップS26）。この結果得られた複素数の実部と虚部は、それぞれBR（1：q／2＋1，p）とBI（1：q／2＋1，p）に格納される。

【0069】次に、BRとBIに格納された配列をブロックレベルで再び転置して、配列Aに格納し（ステップS27）、処理を終了する。このとき、ステップS25と同様のデータ転送により、BR（1：q／2＋1，p）のデータがA（p，1：q／2＋1）に転置され、BI（1：q／2＋1，p）のデータがA（p，q／2＋2：（q／2＋1）×2）に転置される。

【0070】こうして、与えられたp×q個の実数のフーリエ変換の結果のうち、p×（q／2＋1）個が得られる。2次元実フーリエ変換の結果にも1次元実フーリエ変換の場合と同様の共役関係があるため、残りのp×（q－（q／2＋1））個の結果は得られた結果から求められる。

【0071】次に、2次元実FFTの拡張として、3次

元実ＦＦＴの方法について説明する。ｐ×ｑ×ｒの３次元データの離散フーリエ変換は次式で与えられる。

【００７２】

【数７】

$$f_{j1j2j3}=\sum_{n3=0}^{r-1}\sum_{n2=0}^{q-1}\sum_{n1=0}^{p-1}\omega_r^{n3j3}\omega_q^{n2j2}\omega_p^{n1j1}x_{n1n2n3}$$

…（２８）

【００７３】（２８）式は、（２７）式の３次元への拡張となっているため、３次元実ＦＦＴは基本的に２次元実ＦＦＴと同様にして行う。ただし、増えた次元に相当するフーリエ変換処理が追加される。

【００７４】図１８は、ｐ×ｑ×ｒ個の３次元データを（２８）式に従って変換する３次元実ＦＦＴのフローチャートである。図１８において処理が開始されると、まず入出力装置１３は与えられた３次元データを取り込み、それをｐ×ｑ×ｒの３次元配列とする。そして、この３次元配列を第１次元で分割して、各プロセッサに所

$$s1=2\times proc\times k1,$$

$$k1=(p+2\times proc-1)/(2\times proc)\quad\cdots(30)$$

と定め、第１次元をｐｒｏｃ台のプロセッサにより均等に分割したとき、各プロセッサに格納される部分の行数

$$k4=(r/2+1)\times 2$$

以上とする。図１９では、Ａの第３次元の大きさはｋ４となっており、配列ＡはＡ（ｓ１，ｑ，ｋ４）と表され

$$k5=((r/2+proc)/proc)\times proc\quad\cdots(32)$$

として、Ａの第１次元と第３次元の転置用の格納配列Ｂ（２×ｋ５，ｑ，ｓ１）を用意する。Ｂは同じ大きさの２つの配列ＢＲ（ｋ５，ｑ，ｓ１）とＢＩ（ｋ５，ｑ，ｓ１）からなる。ＢＲ、ＢＩの第１次元の大きさｋ５

$$2\times k5\geqq k4=(q/2+1)\times 2$$

なる関係がある。

【００７６】そして、Ａ（ｓ１，ｑ，ｋ４）、ＢＲ（ｋ５，ｑ，ｓ１）、ＢＩ（ｋ５，ｑ，ｓ１）の第１次元を各プロセッサで均等に分割し、分割された各プロセッサの部分をａ（２×ｋ１，ｑ，ｋ４）、ｂｒ（ｋ５／ｐｒｏｃ，ｑ，ｓ１）、ｂｉ（ｋ５／ｐｒｏｃ，ｑ，ｓ１）とする。

【００７７】次に、各プロセッサ内で、１段目の処理を並列に行う。ただし、このときメモリの使用領域を節約するために、各プロセッサに割り当てられた第３次元のベクトルを前半と後半の２組に分け、一方を実部、他方を虚部として、複素数のフーリエ変換を行う。

【００７８】まず、各プロセッサは第３次元のベクトルのペアを作り、一方を実部、他方を虚部として別々の領域に格納する（ステップＳ３２）。具体的には、格納領域ｂｒ、ｂｉをそれぞれ格納配列ｂｒ（ｋ１，ｑ，２×ｋ５）、ｂｉ（ｋ１，ｑ，２×ｋ５）として利用し、ａ（１：ｋ１，ｑ，ｋ４）をｂｒ（ｋ１，ｑ，１：ｋ４）にコピーし、ａ（ｋ１＋１：２×ｋ１，ｑ，ｋ４）をｂｉ（ｋ１，ｑ，１：ｋ４）にコピーする。

定数の第３次元のベクトルを均等に割り当てる（ステップＳ３１）。割当てられた第３次元のベクトルは、各プロセッサのメモリ１６に格納される。

【００７５】図１９は、このときの３次元データを格納する３次元格納配列Ａの分割方法を示している。図１９では、簡単のためプロセッサ数を４（ＰＥ１、ＰＥ２、ＰＥ３、ＰＥ４）としているが、より一般的にはそれをｐｒｏｃと書くことにする。このとき、Ａの第１次元（行の次元）の大きさｓ１を、（２２）、（２３）式と同様に、

…（２９）

…（３０）

が偶数になるようにする。また、Ａの第２次元の大きさはｑとし、第３次元の大きさは、

…（３１）

る。次に、

…（３３）

【００７９】図１９では、４台のプロセッサＰＥ１、ＰＥ２、ＰＥ３、ＰＥ４の各々に割り当てられたａ（２×ｋ１，ｑ，ｋ４）が、前半部分と後半部分に分割されている。ＰＥ１のデータはＲ１とＩ１に分けられ、ＰＥ２のデータはＲ２とＩ２に分けられ、ＰＥ３のデータはＲ３とＩ３に分けられ、ＰＥ４のデータはＲ４とＩ４に分けられる。そして、これらのデータが、図２０に示すように、配列ＢＲ、ＢＩにコピーされる。図２０において、Ｒ１、Ｒ２、Ｒ３、Ｒ４は実部としてＢＲに格納され、Ｉ１、Ｉ２、Ｉ３、Ｉ４は虚部としてＢＩに格納されている。このとき、ＢＲ、ＢＩは、それぞれＢＲ（ｓ１／２，ｑ，２×ｋ５）、ＢＩ（ｓ１／２，ｑ，２×ｋ５）なる配列として利用される。

【００８０】次に、ｂｒ（ｋ１，ｑ，１：ｋ４）の第３次元のベクトルと、対応するｂｉ（ｋ１，ｑ，１：ｋ４）の第３次元のベクトルとを束ねて、各プロセッサで複素数のフーリエ変換を行う（ステップＳ３３）。つまり、ｂｒ（ｋ１，ｑ，１：ｋ４）の１つの第３次元方向のベクトルの各要素を（１２）式のx_{1n}とみなし、対応するｂｉ（ｋ１，ｑ，１：ｋ４）の第３次元方向のベ

クトルの要素を（１２）式のx_{2n}とみなして、複素数z_nに対するフーリエ変換を（１３）式により計算する。ただし、このときQ＝ｒとする。

【００８１】そして、（１６）～（１９）式により、最初の（ｒ／２＋１）個の実フーリエ変換の結果を求め、その実部および虚部をそれぞれａ（２×ｋ１，ｑ，１：ｒ／２＋１）およびａ（２×ｋ１，ｑ，ｒ／２＋２：ｋ４）に格納する（ステップＳ３４）。図２１は、このときの格納方法を示している。図２１において、Ａの第３次元の前半が実部に割り当てられ、後半が虚部に割り当てられていることが分かる。例えばプロセッサＰＥ１内では、図２０のＲ１部分のベクトルの実フーリエ変換結果のうち、最初の（ｒ／２＋１）個が実際に求められ、それらの実部がＣＲ１に格納され、虚部がＤＲ１に格納される。また、図２０のＩ１部分のベクトルの実フーリエ変換結果のうち、最初の（ｒ／２＋１）個が実際に求められ、それらの実部がＣＩ１に格納され、虚部がＤＩ１に格納される。他のプロセッサの領域ＣＲ２、ＤＲ２、ＣＩ２、ＤＩ２、ＣＲ３、ＤＲ３、ＣＩ３、ＤＩ３、ＣＲ４、ＤＲ４、ＣＩ４、ＤＩ４についても同様である。

【００８２】次に、各プロセッサ内で、ａ（２×ｋ１，ｑ，１：ｒ／２＋１）の第２次元方向のベクトルと、ａ（２×ｋ１，ｑ，ｒ／２＋２：ｋ４）の第２次元方向のベクトルとを束ねて、長さｑの複素数のフーリエ変換を行う（ステップＳ３５）。そして、その結果の実部および虚部を、それぞれａ（２×ｋ１，ｑ，１：ｒ／２＋１）およびａ（２×ｋ１，ｑ，ｒ／２＋２：ｋ４）に格納する。

【００８３】次に、２段目の処理を行って、配列Ａのｐ×ｑ×（ｒ／２＋１）×２のデータをブロックに分割し、ブロックレベルで第１次元と第３次元の間の転置を行い、配列Ｂに格納する（ステップＳ３６）。このとき、格納領域ＢＲ、ＢＩはそれぞれ配列ＢＲ（ｋ５，ｑ，ｓ１）、ＢＩ（ｋ５，ｑ，ｓ１）として利用され、その第１次元に関して分割されている。そして、プロセッサ間のデータ転送により、Ａ（ｐ，ｑ，１：ｒ／２＋１）のデータがＢＲ（１：ｒ／２＋１，ｑ，ｐ）に転置され、Ａ（ｐ，ｑ，ｒ／２＋２：（ｒ／２＋１）×２）のデータがＢＩ（１：ｒ／２＋１，ｑ，ｐ）に転置される。この時行われる転置処理は、図７と同様である。ただし、ここでは図７におけるB_{KJ}、A_{JK}^Tは、３次元配列を第１次元および第３次元に関して分割したブロックを意味する。図２２は、転置前の配列Ａの実部または虚部の分割例を示しており、図２３は、転置後のデータを格納する配列ＢＲまたはＢＩの分割例を示している。

【００８４】図２２において、Ａの実部（または虚部）のうちプロセッサＰＥ１内にあるデータは、A_{11}、A_{12}、A_{13}、A_{14}の４つのブロックに分割されている。同様に、プロセッサＰＥ２内のデータはブロックA_{21}、A_{22}、A_{23}、A_{24}に分割され、プロセッサＰＥ３内のデータはブロックA_{31}、A_{32}、A_{33}、A_{34}に分割され、プロセッサＰＥ４内のデータはブロックA_{41}、A_{42}、A_{43}、A_{44}に分割される。また、図２３において、ＢＲ（またはＢＩ）のうちプロセッサＰＥ１内にあるデータは、B_{11}、B_{12}、B_{13}、B_{14}の４つのブロックに分割されている。同様に、プロセッサＰＥ２内のデータはブロックB_{21}、B_{22}、B_{23}、B_{24}に分割され、プロセッサＰＥ３内のデータはブロックB_{31}、B_{32}、B_{33}、B_{34}に分割され、プロセッサＰＥ４内のデータはブロックB_{41}、B_{42}、B_{43}、B_{44}に分割される。

【００８５】図２４は、転置処理を行った後のＢＲ（またはＢＩ）の内容を示している。図２４において、プロセッサＰＥ１内にはA_{11}^T、A_{21}^T、A_{31}^T、A_{41}^Tが格納され、プロセッサＰＥ２内にはA_{12}^T、A_{22}^T、A_{32}^T、A_{42}^Tが格納され、プロセッサＰＥ３内にはA_{13}^T、A_{23}^T、A_{33}^T、A_{43}^Tが格納され、プロセッサＰＥ４内にはA_{14}^T、A_{24}^T、A_{34}^T、A_{44}^Tが格納されている。

【００８６】３次元の場合も、図８に示す２次元配列のデータ転送と同様に、図２２の配列の対角方向に並ぶブロック要素について並列にデータ転送を行う。例えば、図２２の斜線部分のブロックA_{12}、A_{23}、A_{34}、A_{41}の転置データA_{12}^T、A_{23}^T、A_{34}^T、A_{41}^Tは、図２４の斜線部分の各ブロックに並列に転送される。

【００８７】そして、各プロセッサで３段目の処理を並列に行って、配列の転置結果における長さｐの各第３次元のベクトルに対する複素ＦＦＴを計算する（ステップＳ３７）。この結果得られた複素数の実部と虚部は、それぞれＢＲ（１：ｒ／２＋１，ｑ，ｐ）とＢＩ（１：ｒ／２＋１，ｑ，ｐ）に格納される。

【００８８】次に、ＢＲとＢＩに格納された配列の第１次元と第３次元の間の転置を、再びブロックレベルで行って配列Ａに格納し（ステップＳ３８）、処理を終了する。このとき、ステップＳ３６と同様のデータ転送により、ＢＲ（１：ｒ／２＋１，ｑ，ｐ）のデータがＡ（ｐ，ｑ，１：ｒ／２＋１）に転置され、ＢＩ（１：ｒ／２＋１，ｑ，ｐ）のデータがＡ（ｐ，ｑ，ｒ／２＋２：（ｒ／２＋１）×２）に転置される。

【００８９】こうして、与えられたｐ×ｑ×ｒ個の実数のフーリエ変換の結果のうち、ｐ×ｑ×（ｒ／２＋１）個が得られる。３次元実フーリエ変換の結果にも１次元実フーリエ変換の場合と同様の共役関係があるため、残りのｐ×ｑ×（ｒ－（ｒ／２＋１））個の結果は得られた結果から求められる。

【００９０】以上説明したように、本実施例では、ベクトル計算機向けの１～３次元実フーリエ変換を、２つの実数列をそれぞれ実部および虚部とみなした複素フーリエ変換に置き換え、並列計算機の各プロセッサでその結果の約半分を取り出して、最終的にすべての結果を求め

ている。尚、本発明は、１～３次元実フーリエ変換に限らず、より高い次元の離散フーリエ変換にも容易に拡張することができる。例えば、Ｄ次元の実フーリエ変換を計算するには、図１８の３次元実フーリエ変換の処理のステップＳ３５において、（Ｄ－２）次元部分の複素フーリエ変換を続けて行えばよい。

【００９１】

【発明の効果】本発明によれば、メモリ分散型並列計算機システムによる実フーリエ変換処理において、変換される実数配列の１つの次元の大きさが各プロセッサ内で偶数になるように、データが分割配置される。これにより、各プロセッサ内で実数ベクトルのペアを作ることが可能になり、一方を実部、他方を虚部とみなして、他の次元についてのフーリエ変換をそのプロセッサ内で閉じて行うことができる。したがって、並列計算機の並列性およびベクトル性能を効率良く利用できる。

【００９２】また、並列転置と組み合わせることでデータ転送のコストがさらに削減され、処理が効率化される。

【図面の簡単な説明】

【図１】本発明の原理図である。

【図２】実施例の並列計算機システムの構成図である。

【図３】１次元実ＦＦＴのフローチャートである。

【図４】２次元配列の分割を示す図である。

【図５】２次元配列のコピー領域を示す図である。

【図６】実フーリエ変換の結果を２次元配列に格納する方法を示す図である。

【図７】行列の転置処理のフローチャートである。

【図８】転置前の２次元配列を示す図である。

【図９】転置後の２次元配列を示す図である。

【図１０】転置後の２次元配列の内容を示す図である。

【図１１】２次元配列の例を示す図である。

【図１２】コピーされた２次元配列の例を示す図である。

【図１３】複素フーリエ変換後の２次元配列の例を示す図である。

【図１４】実フーリエ変換の結果の格納例を示す図である。

【図１５】ローテーション計算後の２次元配列の例を示す図である。

【図１６】転置後の２次元配列の例を示す図である。

【図１７】２次元実ＦＦＴのフローチャートである。

【図１８】３次元実ＦＦＴのフローチャートである。

【図１９】３次元配列の分割を示す図である。

【図２０】３次元配列のコピー領域を示す図である。

【図２１】実フーリエ変換の結果を３次元配列に格納する方法を示す図である。

【図２２】転置前の３次元配列を示す図である。

【図２３】転置後の３次元配列を示す図である。

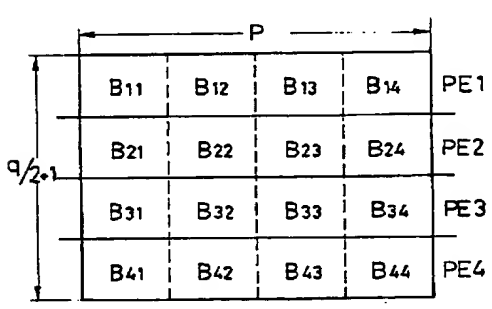
【図２４】転置後の３次元配列の内容を示す図である。

【符号の説明】

1　入力手段
2　配列記憶手段
3　第１の記憶手段
4　第２の記憶手段
5　計算手段
6　転置手段
7　出力手段
１１－１，１１－２，１１－Ｍ　プロセッサ
１２　ネットワーク
１３　入出力装置
１４　通信部
１５　処理部
１６　メモリ
１７　内部バス

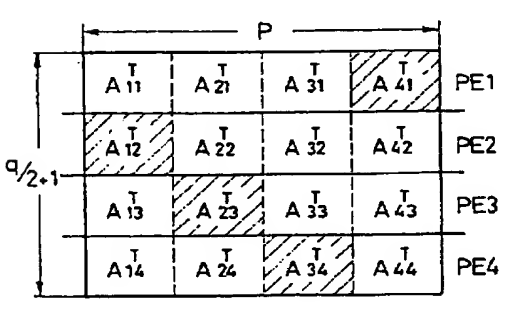
【図９】

転置後の2次元配列を示す図



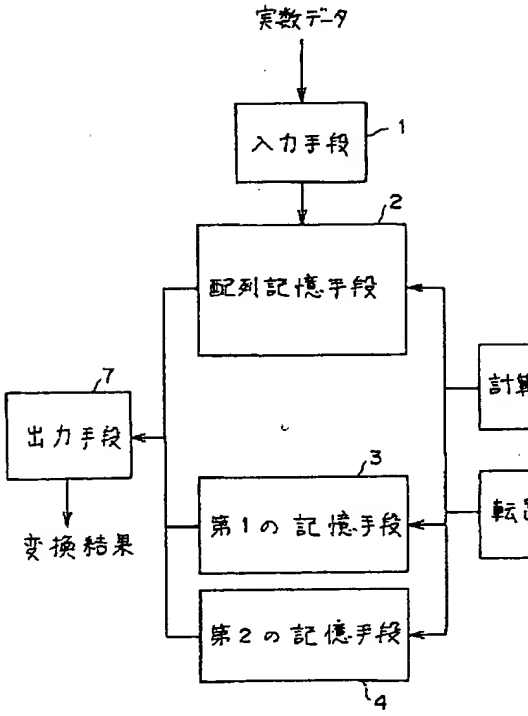
【図１０】

転置後の２次元配列の内容を示す図



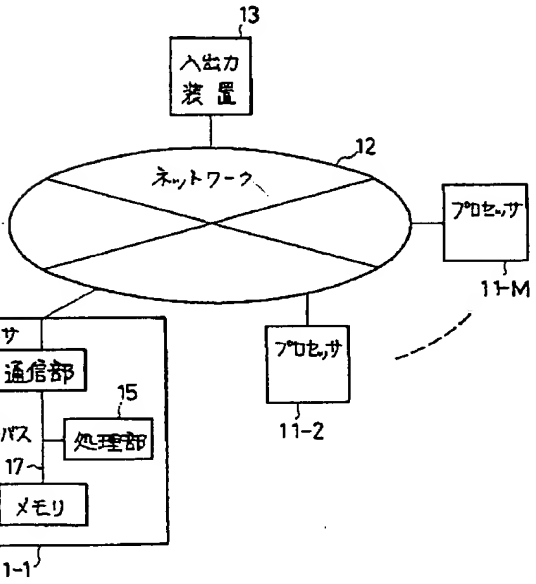
【図１】

本発明の原理図



【図２】

実施例のシステム構成図

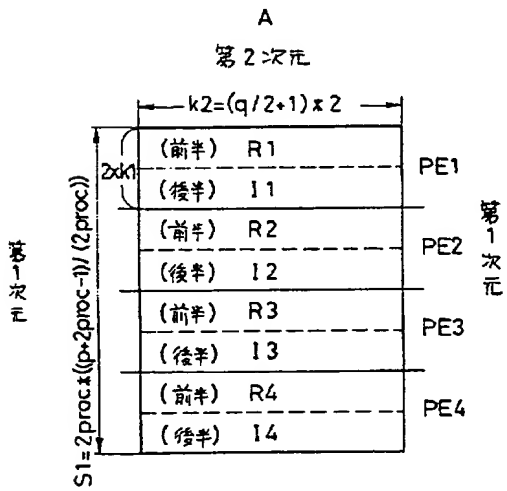


【図８】

転置前の２次元配列を示す図

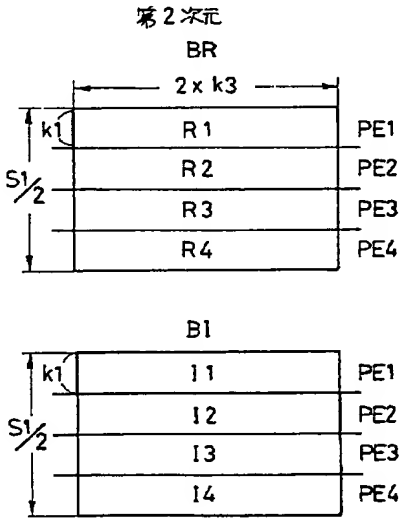
【図４】

２次元配列の分割を示す図

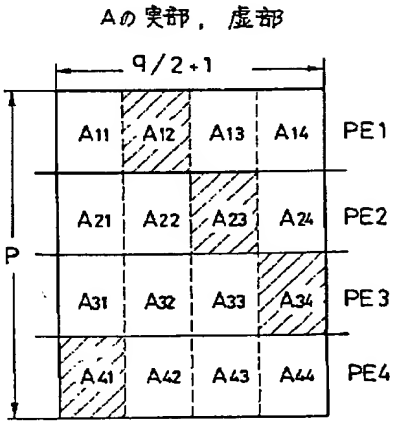


【図５】

２次元配列のコピー領域を示す図

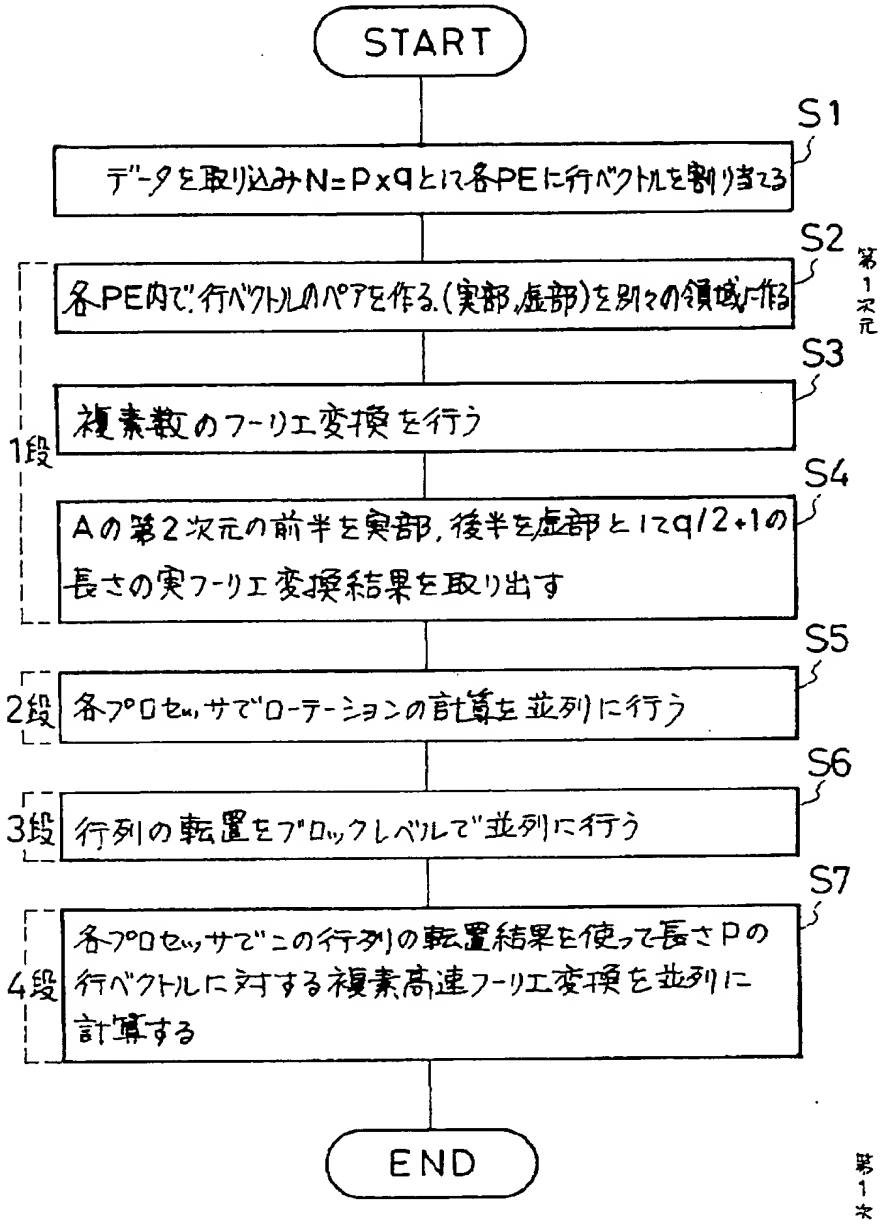


Aの実部，虚部



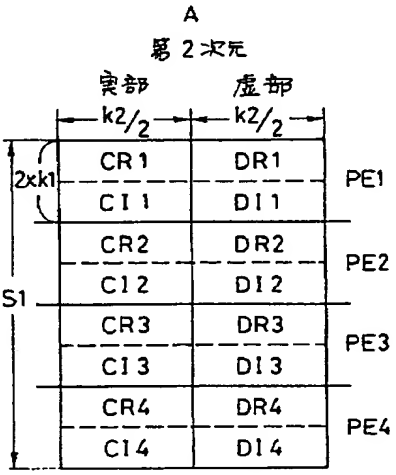
【図３】

一次元実ＦＦＴのフローチャート



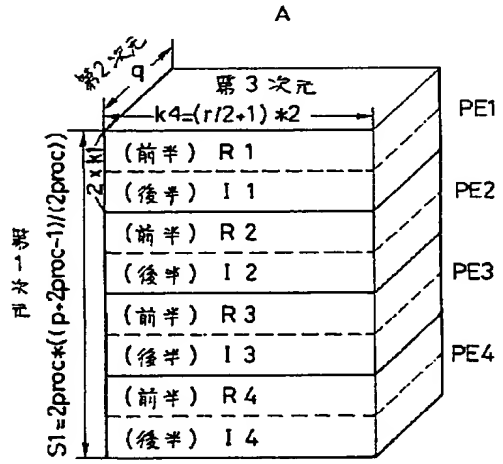
【図６】

実フーリエ変換の結果を２次元配列に格納する方法を示す図



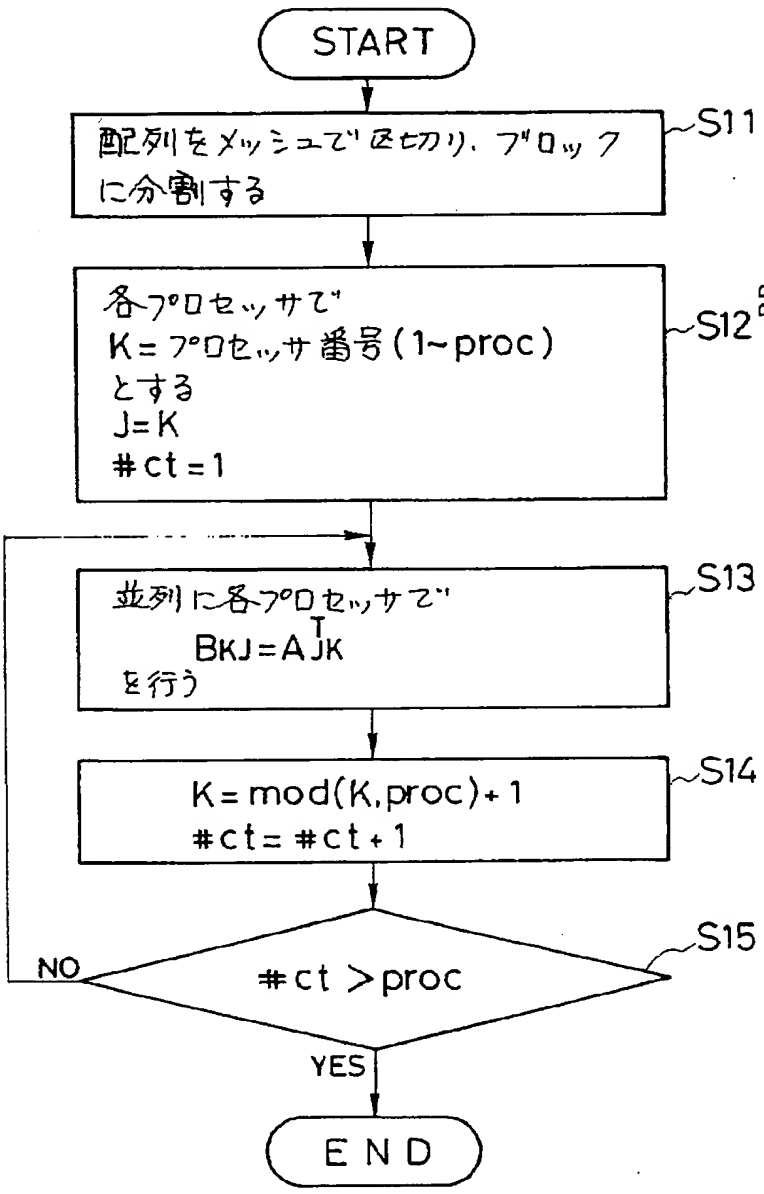
【図１９】

３次元配列の分割を示す図



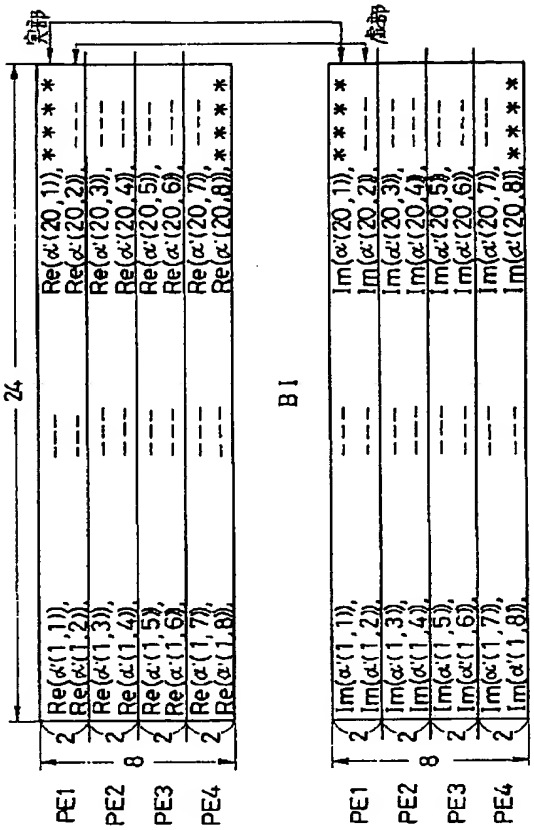
【図７】

行列の転置処理のフローチャート



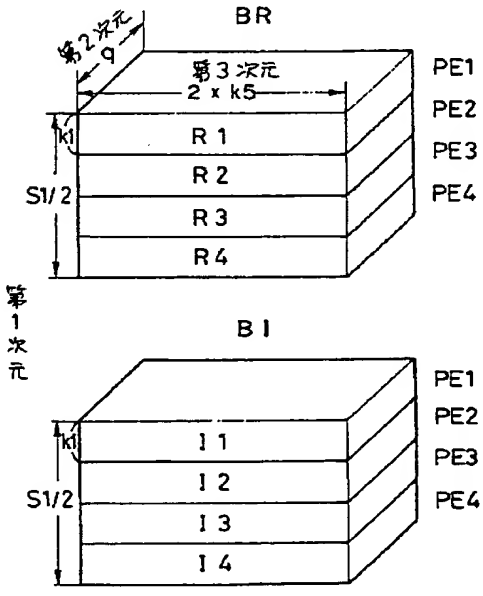
【図１６】

転置後の２次元配列の例を示す図



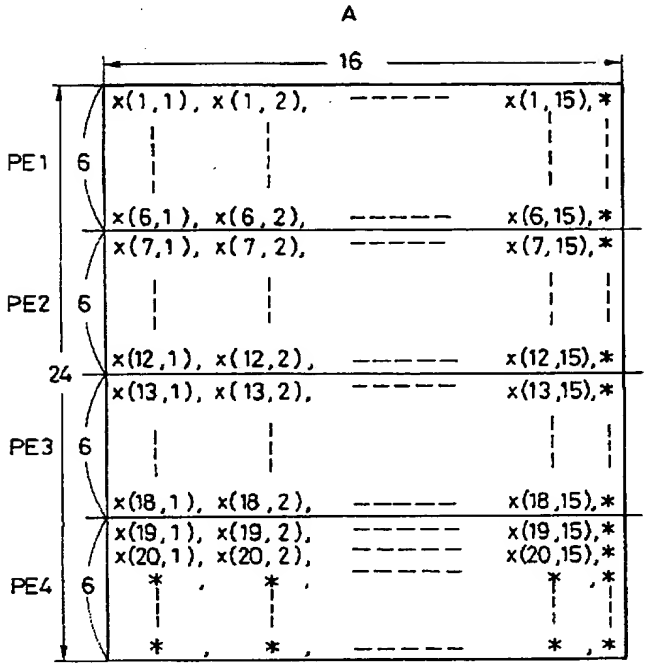
【図２０】

３次元配列のコピー領域を示す図



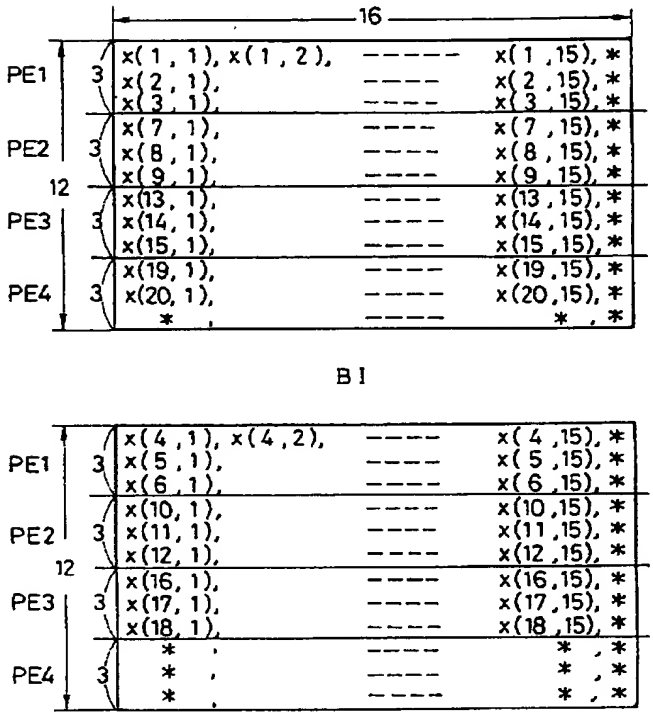
【図１１】

２次元配列の例を示す図



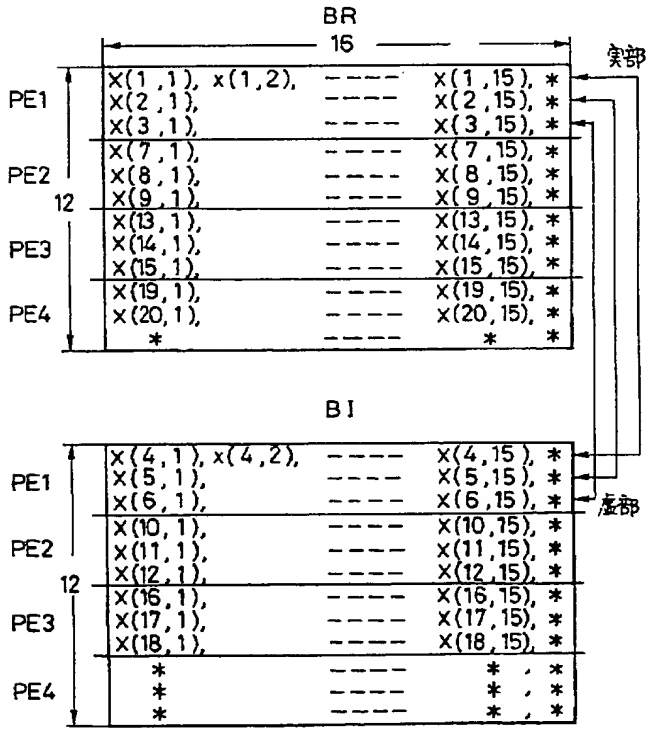
【図１２】

コピーされた２次元配列の例を示す図



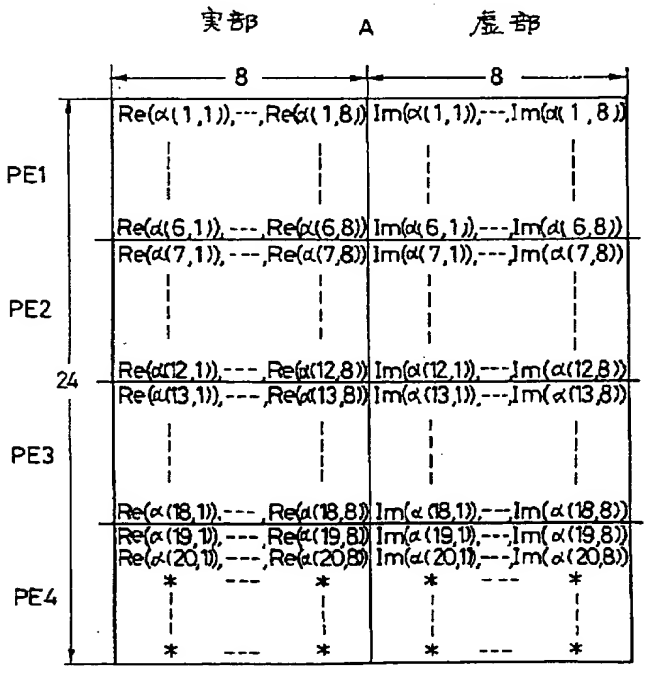
【図１３】

複素フーリエ変換後の２次元配列の例を示す図



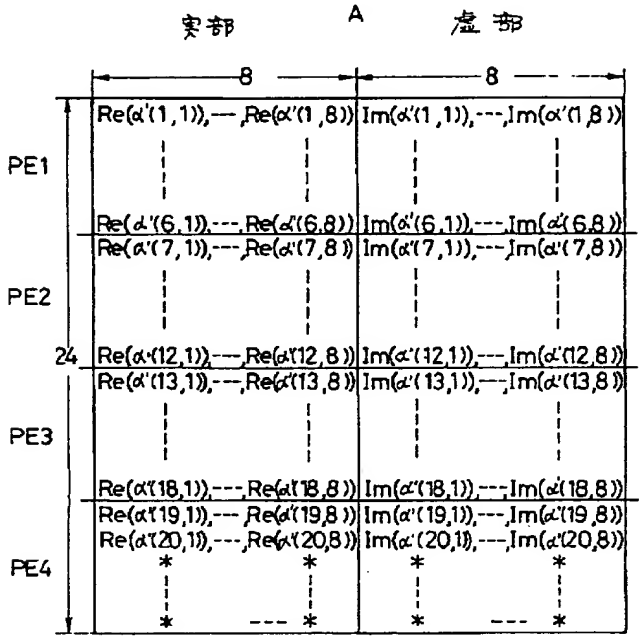
【図１４】

実フーリエ変換の結果の格納例を示す図



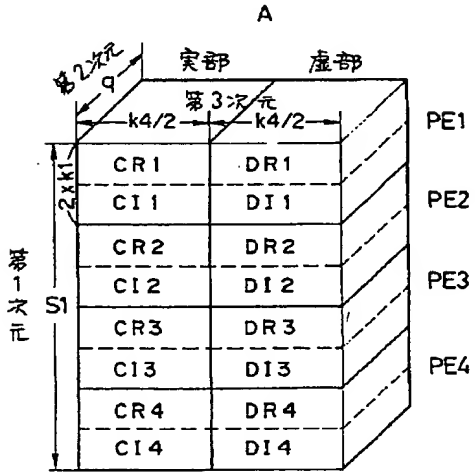
【図１５】

ローテーション計算後の２次元配列の例を示す図



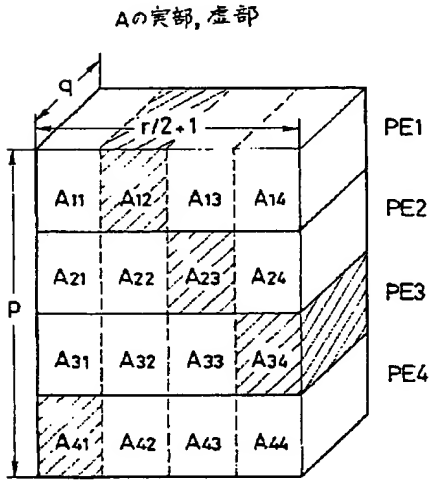
【図２１】

実フーリエ変換の結果を３次元配列に格納する方法を示す図



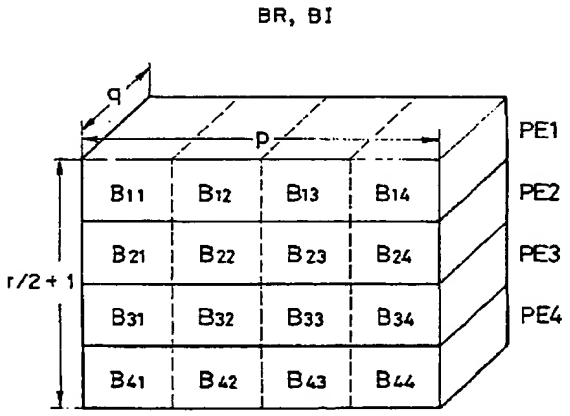
【図２２】

転置前の３次元配列を示す図



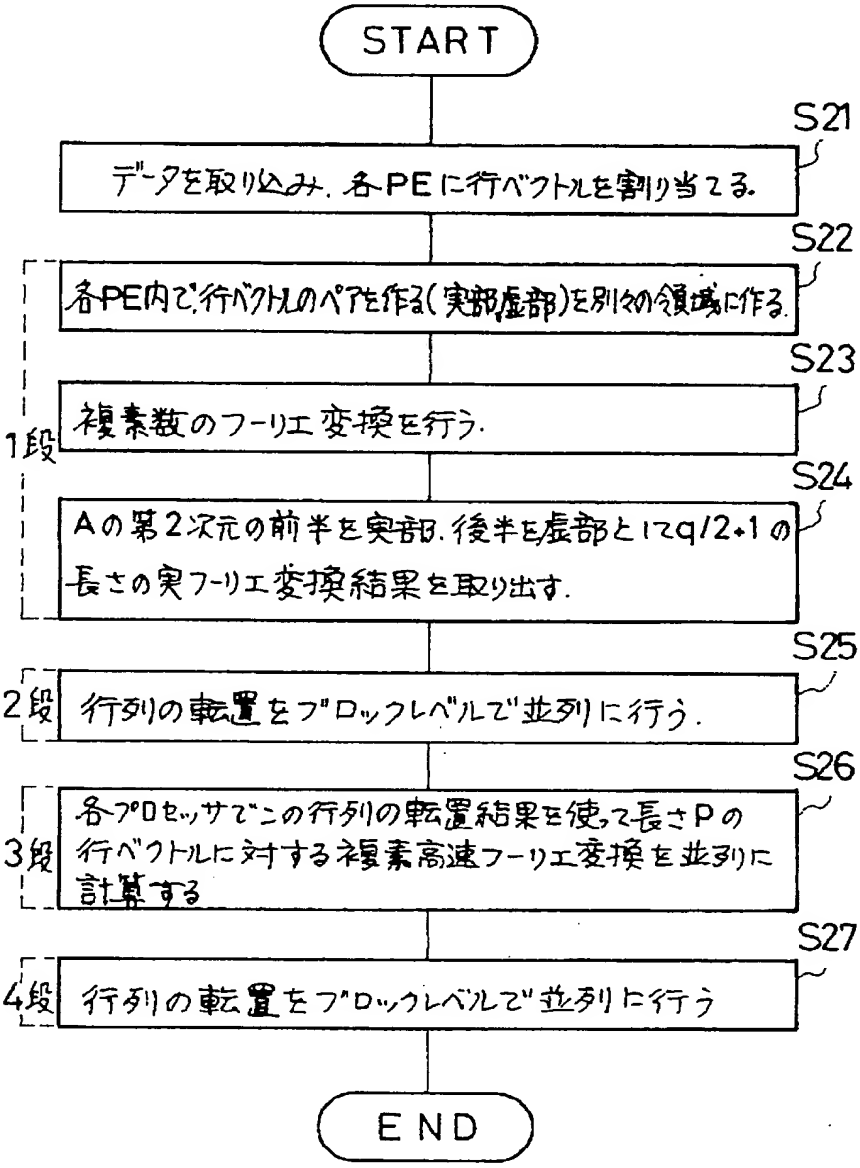
【図２３】

転置後の３次元配列を示す図



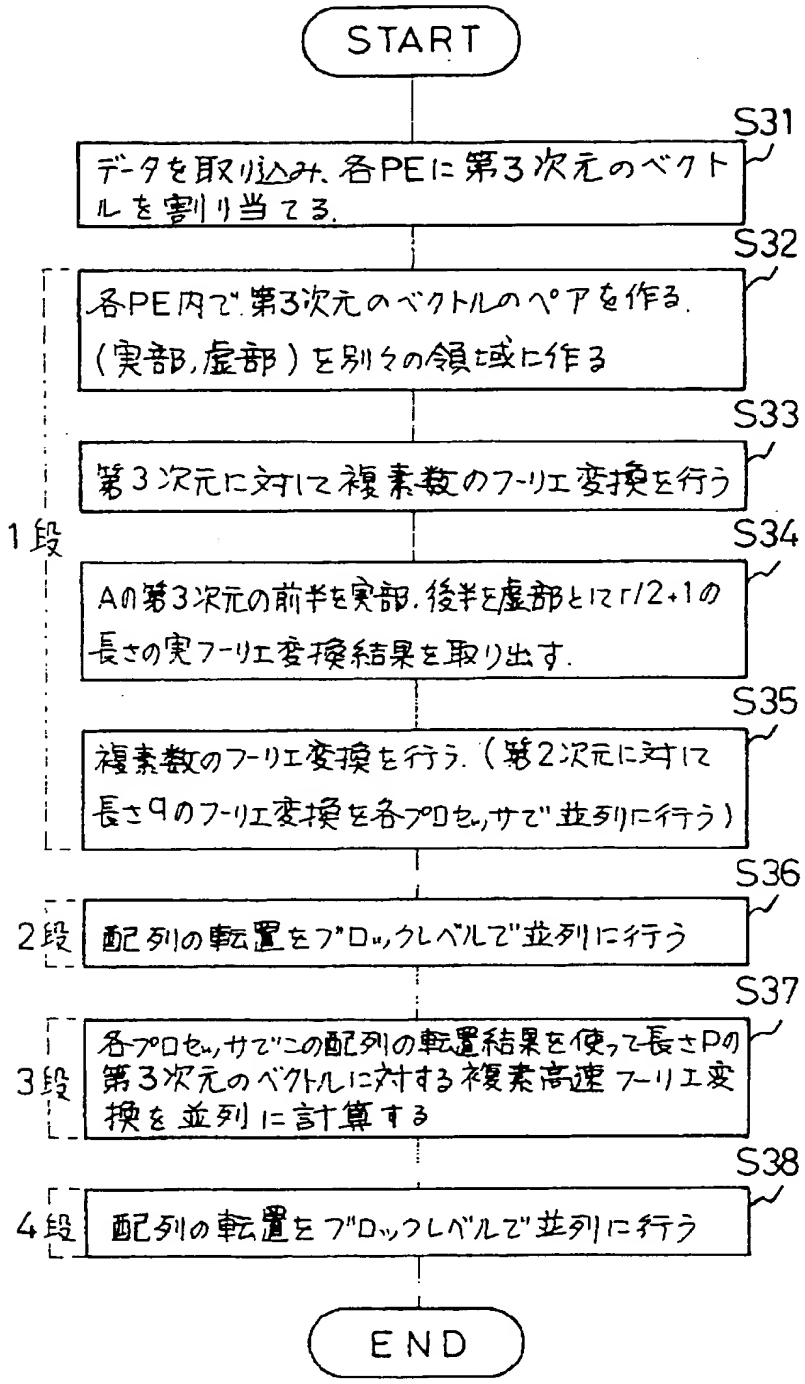
【図17】

2次元実FFTのフローチャート



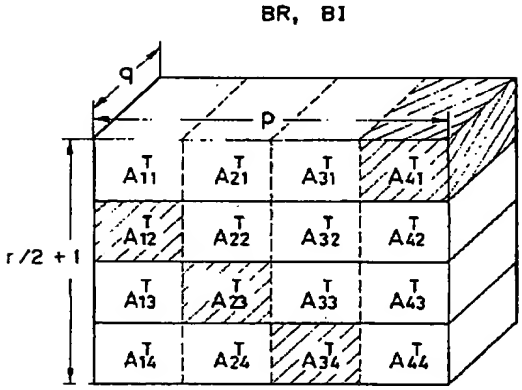
【図18】

3次元実FFTのフローチャート



【図24】

転置後の3次元配列の内容を示す図



フロントページの続き

(72)発明者　マーカス　ヘグランド
オーストラリア国、エーシーティ、2601、
オコーナー、バートストリート　9番地